(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2004-80295
(P2004-80295A)
(43)公開日　平成16年3月11日(2004.3.11)

(51)Int.Cl.[7]		F I			テーマコード（参考）
H04N	7/32	H04N	7/137	Z	5C059
H03M	7/36	H03M	7/36		5J064

審査請求 未請求　請求項の数 9　OL　（全 25 頁）

(21)出願番号　特願2002-236877(P2002-236877)
(22)出願日　平成14年8月15日(2002.8.15)

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番３５号
(74)代理人　100082131
弁理士　稲本　義雄
(72)発明者　近藤　哲二郎
東京都品川区北品川６丁目７番３５号　ソニー株式会社内
(72)発明者　佐藤　浩
東京都品川区北品川６丁目７番３５号　ソニー株式会社内

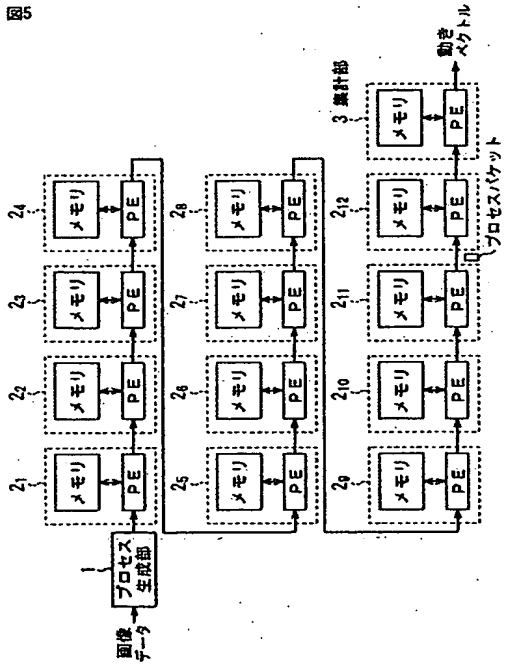
Fターム(参考)　5C059 KK08 KK14 MA00 MA05 NN01 NN28 PP04 TA62 TB08 TC02 TD02 TD05 TD06 TD11 UA02 UA33 UA35 UA36 UA39

最終頁に続く

(54)【発明の名称】動きベクトル検出装置および動きベクトル検出方法

(57)【要約】
【課題】長いデータバスを駆動する際の問題を考慮しないで済む設計容易なハードウェアによって、動きベクトルを検出する。
【解決手段】ＰＥ（Ｐｒｏｃｅｓｓｉｎｇ　Ｅｌｅｍｅｎｔ）とメモリからなる複数の演算処理ユニット2_nを一次元的に接続し、その複数の演算処理ユニット2_nにおいて、命令と必要なデータが配置されたプロセスパケットを転送させることにより、動きベクトルを検出するための処理、即ち、メモリへの画像データの書き込みや、画素どうしの差分絶対値の計算が行われる。本発明は、動きベクトルを検出するＭＰＥＧエンコーダなどに適用することができる。
【選択図】　図５

図5



【特許請求の範囲】
【請求項１】
画像データから、前記画像データの差分に関する差分情報を求め、その差分情報に基づいて、動きベクトルを検出する動きベクトル検出装置であって、
前記動きベクトルを検出する処理を行うためのプロセスを生成し、そのプロセスを実行する命令を含むプロセスデータを出力するプロセス生成手段と、
前記プロセスデータにしたがい、前記プロセスを実行する複数の実行手段と
を備え、
前記複数の実行手段それぞれは、
前記画像データを記憶する記憶手段と、
前記プロセスデータに含まれる命令にしたがい、前記記憶手段への画像データの書き込み、前記記憶手段からの画像データの読み出し、または画像データの前記差分情報の演算のうちのいずれかの処理を行い、その処理結果を、前記プロセスデータに配置して、他の前記実行手段に供給する処理手段と
を有する
ことを特徴とする動きベクトル検出装置。
【請求項２】
前記複数の実行手段は、１次元的に接続されており、
前記複数の実行手段それぞれの前記処理手段は、その後段に接続されている他の前記実行手段に、前記プロセスデータを供給する
ことを特徴とする請求項１に記載の動きベクトル検出装置。
【請求項３】
前記処理手段は、前記プロセスデータに含まれる命令の実行が可能かどうかを判定し、前記命令の実行が可能でない場合、前記プロセスデータを、そのまま、他の前記実行手段に供給する
ことを特徴とする請求項１に記載の動きベクトル検出装置。
【請求項４】
前記処理手段は、前記プロセスデータに含まれる命令が、画像データの前記差分情報の演算を指示する差分情報演算命令である場合、前記差分情報の演算対象である画像データの第１と第２の画素の少なくとも一方が、前記記憶手段に記憶されているかどうかによって、前記差分情報演算命令の実行が可能かどうかを判定する
ことを特徴とする請求項３に記載の動きベクトル検出装置。
【請求項５】
前記第１と第２の画素のいずれも、前記記憶手段に記憶されていない場合、
前記処理手段は、前記プロセスデータを、そのまま、他の前記実行手段に供給する
ことを特徴とする請求項４に記載の動きベクトル検出装置。
【請求項６】
前記第１と第２の画素の両方が、前記記憶手段に記憶されている場合、
前記処理手段は、
前記第１と第２の画素の差分を演算することにより、前記差分情報を求め、
その差分情報を、前記プロセスデータに配置して、他の前記実行手段に供給する
ことを特徴とする請求項４に記載の動きベクトル検出装置。
【請求項７】
前記第１の画素のみが、前記記憶手段に記憶されている場合において、前記第２の画素が前記プロセスデータに配置されているとき、
前記処理手段は、
前記記憶手段に記憶された前記第１の画素と、前記プロセスデータに配置された前記第２の画素との差分を演算することにより、前記差分情報を求め、
その差分情報を前記プロセスデータに配置して、他の前記実行手段に供給する
ことを特徴とする請求項４に記載の動きベクトル検出装置。

【請求項８】
前記第１の画素のみが、前記記憶手段に記憶されている場合において、前記第２の画素が前記プロセスデータに配置されていないとき、
前記処理手段は、
前記記憶手段に記憶された前記第１の画素を、前記プロセスデータに配置して、他の前記実行手段に供給する
ことを特徴とする請求項４に記載の動きベクトル検出装置。
【請求項９】
画像データから、前記画像データの差分に関する差分情報を求め、その差分情報に基づいて、動きベクトルを検出する動きベクトル検出装置の動きベクトル検出方法であって、
前記動きベクトル検出装置は、
前記動きベクトルを検出する処理を行うためのプロセスを生成し、そのプロセスを実行する命令を含むプロセスデータを出力するプロセス生成手段と、
前記プロセスデータにしたがい、前記プロセスを実行する複数の実行手段と
からなり、
前記複数の実行手段それぞれは、
前記画像データを記憶する記憶手段と、
前記プロセスデータに含まれる命令にしたがった処理を行う処理手段と
を有し、
前記プロセスデータに含まれる命令にしたがい、前記記憶手段への画像データの書き込み、前記記憶手段からの画像データの読み出し、または画像データの前記差分情報の演算のうちのいずれかの処理を行う処理ステップと、
その処理結果を、前記プロセスデータに配置して、他の前記実行手段に供給する供給ステップと
を備えることを特徴とする動きベクトル検出方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、動きベクトル検出装置および動きベクトル検出方法に関し、特に、例えば、設計容易なハードウェアによって、動きベクトルを検出することができるようにする動きベクトル検出装置および動きベクトル検出方法に関する。
【０００２】
【従来の技術】
例えば、画像符号化方式としてのＭＰＥＧ（Ｍｏｖｉｎｇ　Ｐｉｃｔｕｒｅ　Ｅｘｐｅｒｔｓ　Ｇｒｏｕｐ）では、ある大きさのブロックのうちの、注目している注目ブロックについて、動きベクトルが検出され、さらに、その動きベクトルに基づいて動き補償が行われることで、注目ブロックの予測画像が求められる。そして、ＭＰＥＧでは、注目ブロックの各画素と、予測画像の対応する画素との差分が演算され、その差分値が符号化されることで、高能率圧縮が実現されている。
【０００３】
動きベクトルの検出アルゴリズムとしては、例えば、ブロックマッチング法が知られている。
【０００４】
例えば、いま、図１に示すように、第ｆフレーム（またはフィールド）のあるブロックを、注目ブロックとするとともに、第ｆ＋１フレームを、動きベクトルの検出のために参照する参照フレームとして、第ｆ＋１フレームから第ｆフレームに向かう動きベクトルを、注目ブロックの動きベクトルとして検出する場合、ブロックマッチング法では、第ｆ＋１フレームの、注目ブロックの位置を中心とする所定の範囲が、動きベクトルの探査を行う探査範囲として設定される。さらに、第ｆ＋１フレームの探査範囲から、注目ブロックと同一の大きさのブロックが、注目ブロックの予測画像の候補である候補ブロックとして選

択され、注目ブロックと候補ブロックとの差分に関する差分情報が求められる。

【０００５】

即ち、例えば、注目ブロックおよび候補ブロックが、横×縦が４×４画素のブロックであるとすると、図２に示すように、注目ブロックの各画素と、候補ブロックの対応する画素との差分が求められ、その差分の絶対値（差分絶対値）が求められる。さらに、その差分絶対値の総和が求められ、探査範囲に選択しうる候補ブロックすべてについて、上述のような差分絶対値の総和が求められる。

【０００６】

そして、探査範囲に選択しうる候補ブロックの中で、差分絶対値の総和を最小にする候補ブロック（以下、適宜、最小候補ブロックという）が求められ、その最小候補ブロックから注目ブロックに向かうベクトルが、注目ブロックの動きベクトルとして求められる。

【０００７】

なお、探査範囲としては、注目ブロックおよび候補ブロックよりも大きな範囲が用いられ、注目ブロックおよび候補ブロックが、上述したように、４×４画素のブロックであるとすると、例えば、３０×３０画素乃至５０×５０画素程度の範囲が、探査範囲として用いられる。

【０００８】

図３は、ブロックマッチング法により動きベクトルを求める、従来の動きベクトル検出装置の一例の構成を示している。

【０００９】

図３の動きベクトル検出装置は、画像データを記憶する画像メモリ１０１と、その画像データを用いて演算を行うことにより動きベクトルを求める動きベクトル抽出部１０２とから構成されており、画像メモリ１０１と、動きベクトル抽出部１０２とは、データバスを介して接続されている。

【００１０】

以上のように構成される動きベクトル検出装置では、画像メモリ１０１に、注目フレームと参照フレームの画像データが記憶される。そして、動きベクトル抽出部１０２は、画像メモリ１０１から、注目ブロックと候補ブロック（の画像データ）を、データバスを介して読み出し、その注目ブロックと候補ブロックとの差分絶対値の総和を求める。さらに、動きベクトル抽出部１０２は、探査範囲内に選択しうる候補ブロックの中で、差分絶対値の総和を最小にする候補ブロック（最小候補ブロック）を求め、その最小候補ブロックから注目ブロックに向かうベクトルを、注目ブロックの動きベクトルとして求めて出力する。

【００１１】

【発明が解決しようとする課題】

図３の動きベクトル検出装置において、動きベクトルを検出する場合、画像メモリ１０１からは、頻繁に、大量の画像データが読み出され、データバスを介して、動きベクトル抽出部１０２に供給される。

【００１２】

一方、画像データを記憶する画像メモリ１０１は、一般に、複数のメモリ（半導体メモリ）で構成される。即ち、図３では、画像メモリ１０１は、６個のメモリ１０１_1乃至１０１_6で構成されている。

【００１３】

画像メモリ１０１を構成するメモリ１０１_1乃至１０１_6それぞれは、比較的広い面積を占有するから、メモリ１０１_1乃至１０１_6それぞれから、動きベクトル抽出部１０２に画像データを転送するには、メモリ１０１_1乃至１０１_6それぞれと動きベクトル抽出部１０２とを結ぶデータバスとしては、比較的長いものが必要となる。そして、長いデータバスを駆動する場合には、種々の問題が生じることとなる。

【００１４】

具体的には、データバスが長い場合、そのデータバスを構成する配線と基板との間の容量

が大になり、データの転送に、大きな遅延（配線遅延）が生じる。さらに、データバスを構成する配線どうしの間に生じる容量によって、配線どうしの間に、クロストークが生じることとなる。そして、近年においては、半導体プロセスの微細化が進行しており、配線の間のクロストークが大きな問題となってきている。

【００１５】

即ち、半導体プロセスの微細化により、配線どうしの間隔が狭くなると、配線の抵抗（配線抵抗）が大となるから、これを防止するために、配線の厚さを厚くする必要がある。そして、配線どうしの間隔が狭くなり、かつ配線の厚さが厚くなると、配線どうしの間の容量が大となり、クロストークを無視することができなくなる。

【００１６】

さらに、従来においては、配線について生じる配線容量については、配線と基板との間の容量を考慮すれば良く、また、基板の電位は一定なので、画像メモリ１０１のシミュレーションを行う場合に、配線容量は、それほど大きな問題とならなかった。

【００１７】

しかしながら、配線容量として、上述のように、配線どうしの間の容量が大となって支配的になると、注目している配線に隣接した配線における信号遷移の仕方によって、注目している配線の見かけの容量が変化し、配線遅延も変化することから、シミュレーションを行うことが困難となる。

【００１８】

また、画像メモリ１０１のデータバスが長いと、その配線端での反射による信号波形の乱れが顕著になる。

【００１９】

そこで、図４に示すように、画像メモリ１０１と動きベクトル抽出部１０２との間に、キャッシュメモリ１０３を設けて、動きベクトル検出装置を構成する方法がある。

【００２０】

図４の動きベクトル検出装置において、キャッシュメモリ１０３は、動きベクトル抽出部１０２で頻繁に用いられる画像データを、画像メモリ１０１から読み出して記憶する。そして、動きベクトル抽出部１０２は、キャッシュメモリ１０３に記憶された画像データを用い、ブロックマッチング法により動きベクトルを求める。

【００２１】

図４の動きベクトル検出装置では、画像メモリ１０１からキャッシュメモリ１０３に転送されて記憶された画像データについては、画像メモリ１０１から読み出す必要がないので、長いデータバスを介して画像メモリ１０１から画像データを読み出す際の、上述した問題の頻度を低減することができる。

【００２２】

しかしながら、この場合、画像メモリ１０１とは別に、キャッシュメモリ１０３という冗長なメモリが必要となり、そのオーバヘッドが問題となる。

【００２３】

本発明は、このような状況に鑑みてなされたものであり、例えば、長いデータバスやキャッシュメモリの不要な設計容易なハードウェアによって、動きベクトルを検出することができるようにするものである。

【００２４】

【課題を解決するための手段】

本発明の動きベクトル検出装置は、動きベクトルを検出する処理を行うためのプロセスを生成し、そのプロセスを実行する命令を含むプロセスデータを出力するプロセス生成手段と、プロセスデータにしたがい、プロセスを実行する複数の実行手段とを備え、複数の実行手段それぞれは、プロセスデータに含まれる命令にしたがい、記憶手段への画像データの書き込み、記憶手段からの画像データの読み出し、または画像データの差分情報の演算のうちのいずれかの処理を行い、その処理結果を、プロセスデータに配置して、他の実行手段に供給する処理手段を有することを特徴とする。

【００２５】
本発明の動きベクトル検出方法は、動きベクトル検出装置が、動きベクトルを検出する処理を行うためのプロセスを生成し、そのプロセスを実行する命令を含むプロセスデータを出力するプロセス生成手段と、プロセスデータにしたがい、プロセスを実行する複数の実行手段とからなり、複数の実行手段それぞれは、プロセスデータに含まれる命令にしたがった処理を行う処理手段を有し、プロセスデータに含まれる命令にしたがい、記憶手段への画像データの書き込み、記憶手段からの画像データの読み出し、または画像データの差分情報の演算のうちのいずれかの処理を行う処理ステップと、その処理結果を、プロセスデータに配置して、他の実行手段に供給する供給ステップとを備えることを特徴とする。
【００２６】
本発明の動きベクトル検出装置および動きベクトル検出方法は、プロセスデータに含まれる命令にしたがい、記憶手段への画像データの書き込み、記憶手段からの画像データの読み出し、または画像データの差分情報の演算のうちのいずれかの処理が行われる。そして、その処理結果が、プロセスデータに配置され、他の実行手段に供給される。
【００２７】
【発明の実施の形態】
図５は、本発明を適用した動きベクトル検出装置の一実施の形態の構成例を示している。
【００２８】
この動きベクトル検出装置においては、例えば、前述したブロックマッチング法により、即ち、画像データから、画像データの差分に関する差分情報としての、例えば、差分絶対値の総和が求められ、その差分情報に基づいて、動きベクトルが検出されるようになっている。
【００２９】
即ち、プロセス生成部１は、動きベクトルを検出する処理を行うためのプロセスを生成し、そのプロセスを実行する命令を含むパケットであるプロセスパケット（プロセスデータ）を出力する。なお、プロセス生成部１には、動きベクトルの検出対象である画像データが供給されるようになっており、プロセス生成部１は、プロセスパケットに、その画像データも、必要に応じて配置する。
【００３０】
プロセス生成部１の後段には、複数の演算処理ユニット2_nが１次元的に接続されている。ここで、図５の実施の形態では、１２の演算処理ユニット2_1乃至2_{12}が一次元的に接続されている。
【００３１】
演算処理ユニット2_nは、前段の演算処理ユニット2_{n-1}から供給されるプロセスパケットを受信し、そのプロセスパケットに含まれる命令にしたがった処理を行う。さらに、演算処理ユニット2_nは、必要に応じて、その処理結果を、プロセスパケットに配置し、後段の演算処理ユニット2_{n+1}に供給する。これにより、最後の演算処理ユニット2_{12}は、後述するように、注目ブロックとある候補ブロックとの差分絶対値の総和を配置したプロセスパケットを出力する。
【００３２】
但し、最初の演算処理ユニット2_1は、プロセスパケットを、プロセス生成部１から受信するようになっている。また、最後の演算処理ユニット2_{12}は、プロセスパケットを、集計部３に供給するようになっている。
【００３３】
集計部３は、最後の演算処理ユニット2_{12}に接続されており、最後の演算処理ユニット2_{12}が出力するプロセスパケットに配置された注目ブロックと候補ブロックとの差分絶対値の総和から、動きベクトルを求めて出力する。
【００３４】
なお、図５の実施の形態では、１２の演算処理ユニット2_1乃至2_{12}が設けられているが、演算処理ユニット2_nの数は、１２に限定されるものではない。

【００３５】
次に、図６は、図５の演算処理ユニット2_nの構成例を示している。
【００３６】
演算処理ユニット2_nは、ＰＥ（Ｐｒｏｃｅｓｓｉｎｇ　Ｅｌｅｍｅｎｔ）１１と、メモリ１２から構成されている。
【００３７】
ＰＥ１１は、フリップフロップ（Ｆ／Ｆ）２１とＡＬＵ（Ａｒｉｔｈｍｅｔｉｃ　ａｎｄ　Ｌｏｇｉｃａｌ　Ｕｎｉｔ）部２２から構成されている。フリップフロップ２１は、そこに供給されるシステムクロックに同期して、前段の演算処理ユニット2_n（またはプロセス生成部１）から供給されるプロセスパケットを受信し、一時記憶する。ＡＬＵ部２２は、フリップフロップ２１に記憶されたプロセスパケットに含まれる命令にしたがった処理を行い、必要に応じて、その処理結果をプロセスパケットに配置して、後段の演算処理ユニット2_{n+1}（または集計部３）に供給する。
【００３８】
ここで、ＡＬＵ部２２は、少なくとも、命令のデコード、デコードした命令の実行、プロセスパケットにおける後述する状態部の書き換え、プロセスパケットの送り出し（プロセスパケットの、後段への供給）の機能を有する。ＡＬＵ部２２は、フリップフロップ２１に配置されたプロセスパケットに含まれる命令をデコードし、その命令の実行が可能であれば、その命令を実行する。さらに、ＡＬＵ部２２は、命令を実行することにより行った処理の結果を、必要に応じて、プロセスパケットに配置するとともに、そのプロセスパケットの状態部を、必要に応じて書き換え、後段に送り出す。
【００３９】
メモリ１２は、ＡＬＵ部２２とデータバスを介して接続されており、ＡＬＵ部２２から供給される画像データを記憶し、また、記憶した画像データを読み出して、ＡＬＵ部２２に供給する。即ち、プロセスパケットに配置される命令の中には、画像データの書き込みを指示する書き込み命令と、画像データの読み出しを指示する読み出し命令がある。そして、ＡＬＵ部２２は、プロセスパケットに、書き込み命令と画像データが配置されている場合には、その画像データを、データバスを介し、メモリ１２に供給して書き込む。また、ＡＬＵ部２２は、プロセスパケットに、読み出し命令が配置されている場合には、メモリ１２から、データバスを介して、画像データを読み出し、プロセスパケットに配置する。
【００４０】
ここで、図６の実施の形態では、メモリ１２は、その記憶領域が５つのバンクに分割されており、これにより、５フィールド（またはフレーム）の画像データを、別々のバンクに記憶することができるようになっている。但し、メモリ１２の各バンクは、１フィールド分の画像データを記憶することができるだけの記憶容量を有している必要はない。即ち、本実施の形態では、メモリ１２の各バンクは、１２の演算処理ユニット2_1乃至2_{12}の合計で、少なくとも探査範囲の画像データを記憶することのできる記憶容量を有していれば良い。つまり、本実施の形態では、１バンクの記憶容量の１２倍の記憶容量が、探査範囲の画像データのデータ量以上であれば良い。従って、メモリ１２としては、図３で説明したような配線容量が問題とならないメモリ、即ち、それほど記憶容量が大きくなく、ＰＥ１１との間のデータバスを短くすることができるメモリを採用することができる。
【００４１】
なお、ここでは、例えば、メモリ１２の１バンクの容量は、その１２倍の容量が１フィールド（またはフレーム）の画像となるように設定されている。
【００４２】
また、図５の実施の形態では、集計部３も、演算処理ユニット2_nと同様に構成されている。即ち、集計部３は、ＰＥ１１とメモリ１２で構成されている。但し、集計部３は、演算処理ユニット2_nと異なる構成とすることが可能であるが、演算処理ユニット2_nと同一構成とした方が、動きベクトル検出装置の製造コストを低コスト化することが可能となる。

【００４３】
次に、図７は、図５のプロセス生成部１が生成し、演算処理ユニット２$_1$乃至２$_{,2}$を順次転送されていくプロセスパケットのフォーマットを示している。
【００４４】
プロセスパケットは、例えば、その先頭から、ＰＩＤ（Ｐｒｏｃｅｓｓ　Ｉｄｅｎｔｉｆｉｃａｔｉｏｎ）部、状態部、命令部、アドレス部、データ部が、順次設けられて構成されている。
【００４５】
ＰＩＤ部には、ＰＩＤが配置される。ここで、ＰＩＤとしては、ある注目ブロックの動きベクトルを求めるまでに行うべき個々のプロセスを区別することができる情報であれば、どのような情報でも採用することが可能である。
【００４６】
即ち、ＰＩＤとしては、例えば、注目ブロックの位置を表すアドレスと、候補ブロックの位置を表すアドレスとの組み合わせなどを採用することが可能である。
【００４７】
また、注目ブロックに対する候補ブロックに、例えば通し番号となるような番号情報を与えておく場合には、ＰＩＤとして、その番号情報を採用することが可能である。なお、ＰＩＤとして、番号情報を採用する場合、集計部３において、すべての通し番号のプロセスパケットが揃うことにより、ある注目ブロックについて、探査範囲で選択可能なすべての候補ブロックとの間の差分絶対値和の総和が得られたことを認識することが可能となる。
【００４８】
状態部は、画像書き込み状態部、画像読み出し状態部、注目ブロック読み出し状態部、候補ブロック読み出し状態部、差分絶対値演算状態部から構成される。
【００４９】
画像書き込み状態部には、メモリ１２に対して、ある画像データの書き込みが、まだ行われていないことを表す状態情報「未」、その画像データの書き込みが行われている途中であることを表す状態情報「中」、その画像データの書き込みが終了したことを表す状態情報「終」のうちのいずれかがセットされる。
【００５０】
即ち、例えば、いま、演算処理ユニット２$_n$を構成するメモリ１２を、メモリ１２$_n$と表すこととして、ある画像データを、例えば、演算処理ユニット２$_1$と２$_{,1}$のメモリ１２$_1$と１２$_{,1}$の２つに分けて書き込む場合を考えると、メモリ１２$_1$と１２$_{,1}$のいずれにも画像データが書き込まれていない場合、画像書き込み状態部には、状態情報「未」がセットされる。また、メモリ１２$_1$と１２$_{,1}$のうちのいずれか一方だけに画像データが書き込まれた場合、画像書き込み状態部には、状態情報「中」がセットされる。さらに、メモリ１２$_1$と１２$_{,1}$の両方に画像データが書き込まれた場合、画像書き込み状態部には、状態情報「終」がセットされる。
【００５１】
なお、状態部にセットされる状態情報は、演算処理ユニット２$_n$に対して、画像データの書き込みなどの状態がどのようになっているかを知らせるメッセージの役割を果たす。
【００５２】
画像読み出し状態部には、メモリ１２からの、ある画像データの読み出しが、まだ行われていないことを表す状態情報「未」、その画像データの読み出しが行われている途中であることを表す状態情報「中」、その画像データの読み出しが終了したことを表す状態情報「終」のうちのいずれかがセットされる。
【００５３】
注目ブロック読み出し状態部には、メモリ１２からの、注目ブロックの画像データの読み出しが、まだ行われていないことを表す状態情報「未」、注目ブロックの画像データの読み出しが行われている途中であることを表す状態情報「中」、注目ブロックの画像データの読み出しが終了したことを表す状態情報「終」のうちのいずれかがセットされる。

【００５４】
候補ブロック読み出し状態部には、メモリ１２からの、候補ブロックの画像データの読み出しが、まだ行われていないことを表す状態情報「未」、候補ブロックの画像データの読み出しが行われている途中であることを表す状態情報「中」、候補ブロックの画像データの読み出しが終了したことを表す状態情報「終」のうちのいずれかがセットされる。
【００５５】
差分絶対値演算状態部には、ある注目ブロックと候補ブロックとの差分絶対値の総和（差分絶対値和）の演算が、まだ行われていないことを表す状態情報「未」、差分絶対値和の演算が行われている途中であることを表す状態情報「中」、差分絶対値和の演算が終了したことを表す状態情報「終」のうちのいずれかがセットされる。
【００５６】
ここで、状態情報は、例えば、２ビットとし、「未」、「中」、「終」には、それぞれ「１１」、「１０」、「００」を割り当てることができる。
【００５７】
命令部には、行うべき処理を指示する命令が配置される。ここで、命令としては、例えば、メモリ１２への画像データの書き込みを指示する書き込み命令、メモリ１２からの画像データの読み出しを指示する読み出し命令、注目ブロックと候補ブロックとの差分絶対値和の演算を指示する差分絶対値和演算命令、各候補ブロックについて求められた差分絶対値和のうちの最小値を求め、その最小値に基づいて、注目ブロックの動きベクトルを求めることを指示する最小値判定命令の４つの命令が、少なくとも用意されている。
【００５８】
なお、命令が、上述の書き込み命令、読み出し命令、差分絶対値和演算命令、および最小値判定命令の４（＝2^2）つである場合には、命令部は、２ビットで十分である。但し、命令部は、将来の命令の種類の拡張も考慮して、２ビットより多い、例えば、４ビットなどとしておくことが望ましい。
【００５９】
アドレス部は、フィールド指定部、読み出しアドレス部、書き込みアドレス部、注目ブロックアドレス部、候補ブロックアドレス部から構成される。
【００６０】
フィールド指定部には、候補ブロックのフィールドを表すフィールド情報がセットされる。即ち、本実施の形態では、図６に示したように、メモリ１２に、５フィールドの画像データが記憶されるが、フィールド指定部には、メモリ１２に記憶された画像データのうち、候補ブロックが存在するフィールドを表すフィールド情報がセットされる。なお、図６の実施の形態では、メモリ１２は、５フィールドの画像データを記憶することから、フィールド情報は、３ビットで十分であるが、将来の拡張を考慮して、３ビットより多い、例えば、４ビットなどとするのが望ましい。
【００６１】
読み出しアドレス部には、メモリ１２から画像データを読み出す場合に、即ち、命令部に、読み出し命令がセットされている場合に、その読み出す画像データのメモリ１２のアドレスが配置される。
【００６２】
書き込みアドレス部には、メモリ１２に画像データを書き込む場合に、即ち、命令部に、書き込み命令がセットされている場合に、その画像データを書き込むメモリ１２のアドレスが配置される。
【００６３】
注目ブロックアドレス部には、注目ブロックのアドレスが配置される。ここで、注目ブロックのアドレスとしては、例えば、その注目ブロックのフィールド（注目フィールド）における、注目ブロックの左上の画素の位置を表す位置情報を採用することができる。従って、例えば、１フィールドが、７２０×２４０画素で構成されるとともに、注目ブロックが、４×２画素で構成されるものとすると、注目ブロックのアドレスは、２１６００（＝

７２０／４×２４０／２）通りが存在するから、１５ビットで表すことができる。

【００６４】

候補ブロックアドレス部には、候補ブロックのアドレスが配置される。ここで、候補ブロックのアドレスとしては、例えば、その候補ブロックのフィールド（参照フィールド）における、候補ブロックの左上の画素の位置を表す位置情報を採用することができる。従って、例えば、上述のように、１フィールドが、７２０×２４０画素で構成されるものとすると、候補ブロックのアドレスは、１７２８００（＝７２０×２４０）通りが存在するから、１８ビットで表すことができる。

【００６５】

データ部は、注目ブロックデータ部、候補ブロックデータ部、差分絶対値和部、候補ベクトル部から構成される。

【００６６】

注目ブロックデータ部には、注目ブロックの画像データ、即ち、注目ブロックを構成する画素（の画素値）が配置される。従って、例えば、上述のように、注目ブロックが、４×２画素で構成されるものとし、また、１画素に、例えば、８ビットが割り当てられているとした場合には、注目ブロックデータ部は、６４（＝４×２×８）ビットで構成されることになる。

【００６７】

候補ブロックデータ部には、候補ブロックの画像データ、即ち、候補ブロックを構成する画素が配置される。従って、候補ブロックが、例えば、上述のように、注目ブロックと同様の４×２画素で構成されるものとし、また、１画素に、例えば、８ビットが割り当てられているとした場合には、候補ブロックデータ部は、注目ブロックデータ部と同様に、６４（＝４×２×８）ビットで構成されることになる。

【００６８】

差分絶対値和部には、注目ブロックと候補ブロックの差分絶対値和が配置される。ここで、上述のように、例えば、１画素に８ビットが割り当てられている場合、注目ブロックのある画素と、その画素に対応する候補ブロックの画素値の差分絶対値は、９ビットで表されることになる。

【００６９】

候補ベクトル部には、候補ブロックから注目ブロックに向かうベクトルが、注目ブロックの動きベクトルの候補（候補ベクトル）として配置される。なお、候補ベクトルは、注目ブロックアドレス部に配置された注目ブロックのアドレスと、候補ブロックアドレス部に配置された候補ブロックのアドレスとから求めることが可能である。また、例えば、いま、探査範囲を、６３×６３画素以下とすると、候補ベクトルのｘ方向（横方向）成分と、ｙ方向（縦方向）成分は、いずれも、６ビットで表すことができるから、候補ベクトル部は、１２（＝６＋６）ビットで構成することができる。

【００７０】

次に、図８のフローチャートを参照して、図５のプロセス生成部１が、ある１つのブロックを注目ブロックとして、その注目ブロックの動きベクトルを検出するのに行う処理（プロセス生成処理）について説明する。

【００７１】

まず最初に、ステップＳ１において、プロセス生成部１は、注目ブロックが存在する注目フィールドのうちの注目ブロックの画像データと、候補ブロックが存在する参照フィールドのうちの探査範囲の画像データとを、少なくとも、メモリ１２に書き込む書き込みプロセスの一部または全部を生成し、その書き込みプロセスを実行する書き込み命令を含むプロセスパケットを生成して、ステップＳ２に進む。

【００７２】

なお、プロセス生成部１は、ステップＳ１において、図７に示したプロセスパケットの命令部に書き込み命令を配置する他、メモリ１２に書き込む注目ブロックや候補ブロックの画像データを、注目ブロックデータ部や候補ブロックデータ部に配置するとともに、その

画像データを書き込むメモリ１２のアドレスを、書き込みアドレス部に配置する。さらに、プロセス生成部１は、必要に応じて、プロセスパケットの状態部に、状態情報を配置する。

【００７３】

ステップＳ２では、プロセス生成部１は、直前のステップＳ１で生成したプロセスパケットを、最初の演算処理ユニット２，に送信して、ステップＳ３に進む。

【００７４】

ステップＳ３では、プロセス生成部１は、必要な画像データ、即ち、ここでは、注目ブロックの画像データと、候補ブロックが存在する参照フィールドのうちの探査範囲の画像データとを、少なくとも、メモリ１２に書き込むためのプロセスすべてを生成したかどうかを判定する。ステップＳ３において、必要な画像データをメモリ１２に書き込むためのプロセスすべてを、まだ生成していないと判定された場合、ステップＳ１に戻り、プロセス生成部１は、必要な画像データをメモリ１２に書き込むためのプロセスのうち、まだ書き込まれていない画像データを書き込むためのプロセスを生成し、以下、同様の処理を繰り返す。

【００７５】

また、ステップＳ３において、必要な画像データをメモリ１２に書き込むためのプロセスすべてを生成したと判定された場合、ステップＳ４に進み、プロセス生成部１は、注目ブロックの探索範囲内に選択しうる候補ベクトルすべてについて求められた差分絶対値和のうちの最小値を求め、その最小値に基づいて、注目ブロックの動きベクトルを求める最小値判定プロセスを生成し、その最小値判定プロセスを実行する最小値判定命令を含むプロセスパケットを生成して、ステップＳ５に進む。

【００７６】

ステップＳ５では、プロセス生成部１は、ステップＳ４で生成したプロセスパケットを、最初の演算処理ユニット２，に送信し、ステップＳ６に進む。

【００７７】

ステップＳ６では、プロセス生成部１は、探査範囲内に選択しうるある候補ブロックから、注目ブロックへのベクトル（候補ベクトル）について、注目ブロックと候補ブロックとの差分絶対値和を求めるための差分絶対値和演算プロセスを生成し、その差分絶対値和演算プロセスを実行する差分絶対値和演算命令を含むプロセスパケットを生成して、ステップＳ７に進む。

【００７８】

なお、プロセス生成部１は、ステップＳ６において、図７に示したプロセスパケットの命令部に差分絶対値和演算命令を配置する他、メモリ１２に書き込まれた注目ブロックや候補ブロックのアドレスを、注目ブロックアドレス部や候補ブロックアドレス部に配置するとともに、候補ベクトルを、候補ベクトル部に配置する。また、プロセス生成部１は、候補ブロックのフィールドを表すフィールド情報を、フィールド指定部に配置するとともに、必要に応じて、プロセスパケットの状態部に、状態情報を配置する。

【００７９】

ステップＳ７では、プロセス生成部１は、直前のステップＳ６で生成したプロセスパケットを、最初の演算処理ユニット２，に送信して、ステップＳ８に進む。

【００８０】

ステップＳ８では、プロセス生成部１は、注目ブロックの探索範囲内に選択しうる候補ベクトルすべてについて、差分絶対値和を演算するための差分絶対値和演算プロセスを生成したかどうかを判定する。

【００８１】

ステップＳ８において、注目ブロックの探索範囲内に選択しうる候補ベクトルすべてについて、差分絶対値和を演算するための差分絶対値和演算プロセスを、まだ生成していないと判定された場合、ステップＳ６に戻り、プロセス生成部１は、まだ差分絶対値和演算プロセスを生成していない候補ベクトルについて、差分絶対値和を演算するための差分絶対

値和演算プロセスを生成し、以下、同様の処理を繰り返す。

【００８２】

また、ステップＳ８において、注目ブロックの探索範囲内に選択しうる候補ベクトルすべてについて、差分絶対値和を演算するための差分絶対値和演算プロセスを生成したと判定された場合、処理を終了する。

【００８３】

次に、図９のフローチャートを参照して、図５の演算処理ユニット2_nが、プロセスパケットを受信した場合に行う処理（プロセス実行処理）について説明する。

【００８４】

演算処理ユニット2_nのフリップフロップ２１でプロセスパケットが受信されて記憶されると、ステップＳ１１において、ＡＬＵ部２２は、そのプロセスパケットに配置された命令をデコードし、ステップＳ１２に進む。ステップＳ１２では、ＡＬＵ部２２は、ステップＳ１１でデコードした命令の実行が可能かどうかを判定する。

【００８５】

ステップＳ１２において、命令の実行が可能でないと判定された場合、ステップＳ１３およびＳ１４をスキップして、ステップＳ１５に進み、演算処理ユニット2_nは、フリップフロップ２１に記憶されたプロセスパケットを、そのまま、後段の演算処理ユニット2_{n+1}（または集計部３）に送信し（供給し）、処理を終了する。

【００８６】

また、ステップＳ１２において、命令の実行が可能であると判定された場合、ステップＳ１３に進み、ＡＬＵ部２２は、プロセスパケットに配置された命令を実行し、これにより、その命令にしたがった処理を行う。さらに、ステップＳ１３では、ＡＬＵ部２２は、プロセスパケットに配置された命令にしたがった処理を行うことにより得られたデータを、必要に応じて、プロセスパケットに配置し、ステップＳ１４に進む。

【００８７】

ステップＳ１４では、ＡＬＵ部２２は、ステップＳ１３で行った処理に対応して、必要に応じて、プロセスパケットの状態部を書き換え、ステップＳ１５に進む。ステップＳ１５では、ＡＬＵ部２２は、プロセスパケットを、後段の演算処理ユニット2_{n+1}に送信し、処理を終了する。

【００８８】

次に、図１０のフローチャートを参照して、図５の演算処理ユニット2_nが、書き込み命令を有するプロセスパケットを受信した場合に行う図９のプロセス実行処理について説明する。

【００８９】

なお、この場合、プロセス生成部１が出力するプロセスパケットには、書き込み命令の他、メモリ１２に書き込むべき画像データと、その画像データをメモリ１２に書き込む書き込みアドレスが、少なくとも配置されているものとする。

【００９０】

演算処理ユニット2_nのフリップフロップ２１でプロセスパケットが受信されて記憶されると、ステップＳ１１において、ＡＬＵ部２２は、そのプロセスパケットに配置された書き込み命令をデコードし、ステップＳ１２に進む。ステップＳ１２では、ＡＬＵ部２２は、ステップＳ１１でデコードした書き込み命令の実行が可能かどうか、即ち、ここでは、プロセスパケットに配置されている書き込みアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスかどうかを判定する。

【００９１】

ステップＳ１２において、プロセスパケットに配置されている書き込みアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスでないと判定された場合、即ち、書き込みアドレスが、他の演算処理ユニット2_nが有するメモリ12_nのアドレスである場合、ステップＳ１３およびＳ１４をスキップして、ステップＳ１５に進み、演算処理ユニット2_nは、フリップフロップ２１に記憶されたプロセスパケットを、後段の演算処理

ユニット2_{n+1}に送信し、処理を終了する。

【００９２】

なお、ここでは、ステップＳ１２において、プロセスパケットに配置されている書き込みアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスかどうかを判定する他、プロセスパケットの画像書き込み状態部の状態情報が「終」になっていないかどうかも判定することにより、書き込み命令の実行が可能かどうかが判定される。即ち、プロセスパケットの画像書き込み状態部の状態情報が「終」になっている場合は、プロセスパケットに配置された画像データの書き込みが終了しているということであるから、書き込み命令の実行が可能でないと判定される。

【００９３】

一方、ステップＳ１２において、プロセスパケットに配置されている書き込みアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスであると判定された場合、ステップＳ１３に進み、ＡＬＵ部２２は、プロセスパケットに配置された書き込み命令を実行し、これにより、プロセスパケットに配置されている画像データを、メモリ12_nに書き込み、ステップＳ１４に進む。

【００９４】

ステップＳ１４では、ＡＬＵ部２２は、ステップＳ１３で行った処理に対応して、必要に応じて、プロセスパケットの状態部を書き換え、ステップＳ１５に進む。ステップＳ１５では、ＡＬＵ部２２は、プロセスパケットを、後段の演算処理ユニット2_{n+1}に送信し、処理を終了する。

【００９５】

図８で説明したプロセス生成処理では、プロセス生成部１が、ステップＳ１乃至Ｓ３において、書き込み命令を有するプロセスパケット（以下、適宜、書き込みプロセスパケットという）を生成して、演算処理ユニット2_1に送信するが、この書き込みプロセスパケットが、演算処理ユニット2_1乃至2_{12}を順次転送されることにより、図１１に示すように、少なくとも、注目ブロックと候補ブロックの画像データが、メモリ１２に書き込まれる。

【００９６】

ここで、図１１Ａに示すように、注目ブロックの画像データを影を付して表すとともに、候補ブロックの画像データを斜線を付して表すこととすると、図１１Ｂでは、注目ブロックの一部がメモリ12_1に、他の一部がメモリ12_2に、さらに他の一部がメモリ12_5に、残りがメモリ12_6に、それぞれ書き込まれ、候補ブロックの一部がメモリ12_3に、他の一部がメモリ12_4に、さらに他の一部がメモリ12_7に、残りがメモリ12_8に、それぞれ書き込まれている。

【００９７】

次に、図１２のフローチャートを参照して、図５の演算処理ユニット2_nが、読み出し命令を有するプロセスパケットを受信した場合に行う図９のプロセス実行処理について説明する。

【００９８】

なお、この場合、プロセス生成部１が出力するプロセスパケットには、読み出し命令の他、メモリ１２から読み出すべき画像データの読み出しアドレスが、少なくとも配置されているものとする。

【００９９】

演算処理ユニット2_nのフリップフロップ２１でプロセスパケットが受信されて記憶されると、ステップＳ１１において、ＡＬＵ部２２は、そのプロセスパケットに配置された読み出し命令をデコードし、ステップＳ１２に進む。ステップＳ１２では、ＡＬＵ部２２は、ステップＳ１１でデコードした読み出し命令の実行が可能かどうか、即ち、ここでは、プロセスパケットに配置されている読み出しアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスかどうかを判定する。

【０１００】

ステップＳ１２において、プロセスパケットに配置されている読み出しアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスでないと判定された場合、即ち、読み出しアドレスが、他の演算処理ユニット2_nが有するメモリ12_nのアドレスである場合、ステップＳ１３およびＳ１４をスキップして、ステップＳ１５に進み、演算処理ユニット2_nは、フリップフロップ２１に記憶されたプロセスパケットを、後段の演算処理ユニット2_{n+1}に送信し、処理を終了する。

【０１０１】

なお、ここでは、ステップＳ１２において、プロセスパケットに配置されている読み出しアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスかどうかを判定する他、プロセスパケットの画像読み出し状態部の状態情報が「終」になっていないかどうかも判定することにより、読み出し命令の実行が可能かどうかが判定される。即ち、プロセスパケットの画像読み出し状態部の状態情報が「終」になっている場合は、プロセスパケットに配置された画像データの読み出しが終了しているということであるから、読み出し命令の実行が可能でないと判定される。

【０１０２】

一方、ステップＳ１２において、プロセスパケットに配置されている読み出しアドレスが、演算処理ユニット2_nが有するメモリ12_nのアドレスであると判定された場合、ステップＳ１３に進み、ＡＬＵ部２２は、プロセスパケットに配置された読み出し命令を実行し、これにより、メモリ12_nから画像データを読み出す。さらに、ＡＬＵ部２２は、メモリ12_nから読み出した画像データを、プロセスパケットに配置し、ステップＳ１４に進む。

【０１０３】

ステップＳ１４では、ＡＬＵ部２２は、ステップＳ１３で行った処理に対応して、必要に応じて、プロセスパケットの状態部を書き換え、ステップＳ１５に進む。ステップＳ１５では、ＡＬＵ部２２は、プロセスパケットを、後段の演算処理ユニット2_{n+1}に送信し、処理を終了する。

【０１０４】

次に、図１３のフローチャートを参照して、図５の演算処理ユニット2_nが、差分絶対値和演算命令を有するプロセスパケットを受信した場合に行う図９のプロセス実行処理について説明する。

【０１０５】

なお、この場合、メモリ１２には、既に、差分絶対値和の演算対象である注目ブロックと候補ブロックの画像データが少なくとも書き込まれているものとする。さらに、プロセス生成部１が出力するプロセスパケットには、差分絶対値和演算命令の他、メモリ１２に記憶された注目ブロックと候補ブロックのアドレス、候補ブロックから注目ブロックに向かう候補ベクトルが、少なくとも配置されているものとする。

【０１０６】

演算処理ユニット2_nのフリップフロップ２１でプロセスパケットが受信されて記憶されると、ステップＳ１１において、ＡＬＵ部２２は、そのプロセスパケットに配置された差分絶対値和演算命令をデコードし、ステップＳ１２に進む。ステップＳ１２では、ＡＬＵ部２２は、ステップＳ１１でデコードした差分絶対値和演算命令の実行が可能かどうか、即ち、ここでは、注目ブロックまたは候補ブロックのうちの少なくとも一方の画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されているかどうかを判定する。

【０１０７】

ここで、注目ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されているかどうかは、プロセスパケットに配置された注目ブロックのアドレスから判定することができる。候補ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されているかどうかも、プロセスパケットに配置された候補ブロックのアドレスから判定することができる。

【０１０８】

ステップＳ１２において、注目ブロックおよび候補ブロックのうちのいずれも、演算処理ユニット2_nが有するメモリ12_nに記憶されていないと判定された場合、ステップＳ１３およびＳ１４をスキップして、ステップＳ１５に進み、演算処理ユニット2_nは、フリップフロップ２１に記憶されたプロセスパケットを、後段の演算処理ユニット2_{n+1}に送信し、処理を終了する。

【０１０９】

なお、ここでは、ステップＳ１２において、注目ブロックまたは候補ブロックのうちの少なくとも一方の画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されているかどうかを判定する他、プロセスパケットの画像差分絶対値演算状態部の状態情報が「終」になっていないかどうかも判定することにより、差分絶対値和演算命令の実行が可能かどうかが判定される。即ち、プロセスパケットの画像差分絶対値演算状態部の状態情報が「終」になっている場合は、注目ブロックと候補ブロックとの差分絶対値和演算が終了しているということであるから、差分絶対値和演算命令の実行が可能でないと判定される。

【０１１０】

一方、ステップＳ１２において、注目ブロックまたは候補ブロックのうちの少なくとも一方の画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されていると判定された場合、ステップＳ１３に進み、ＡＬＵ部２２は、プロセスパケットに配置された差分絶対値和演算命令を実行する。

【０１１１】

即ち、ステップＳ１３において、差分絶対値和演算命令が実行される場合においては、まず最初に、ステップＳ２１において、ＡＬＵ部２２が、注目ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されているかどうかを判定する。ステップＳ２１において、注目ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されていないと判定された場合、ステップＳ２２をスキップして、ステップＳ２３に進む。

【０１１２】

また、ステップＳ２１において、注目ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されていると判定された場合、ステップＳ２２に進み、ＡＬＵ部２２は、メモリ12_nに記憶されている注目ブロックの画素を読み出し、プロセスパケットに配置して、ステップＳ２３に進む。

【０１１３】

ステップＳ２３では、ＡＬＵ部２２が、候補ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されているかどうかを判定する。ステップＳ２３において、候補ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されていないと判定された場合、ステップＳ２４をスキップして、ステップＳ２５に進む。

【０１１４】

また、ステップＳ２３において、候補ブロックの画素が、演算処理ユニット2_nが有するメモリ12_nに記憶されていると判定された場合、ステップＳ２４に進み、ＡＬＵ部２２は、メモリ12_nに記憶されている候補ブロックの画素を読み出し、プロセスパケットに配置して、ステップＳ２５に進む。

【０１１５】

ステップＳ２５では、ＡＬＵ部２２が、差分絶対値和の計算が可能かどうかを判定する。即ち、ステップＳ２５では、プロセスパケットに、注目ブロックの画素が配置されており、かつ、その注目ブロックの画素に対応する候補ブロックの画素も配置されているかどうかによって、差分絶対値和の計算が可能かどうかが判定される。

【０１１６】

ステップＳ２５において、差分絶対値和の計算が可能でないと判定された場合、即ち、プロセスパケットに、注目ブロックの画素が配置されていないか、または配置されていても、その注目ブロックの画素に対応する候補ブロックの画素が配置されていない場合、ステップＳ２６をスキップし、これにより、差分絶対値和演算命令の実行を終了して、ステッ

プＳ１４に進む。

【０１１７】

また、ステップＳ２５において、差分絶対値和の計算が可能であると判定された場合、即ち、プロセスパケットに、注目ブロックの画素が配置されており、かつ、その注目ブロックの画素に対応する候補ブロックの画素も配置されている場合、ステップＳ２６に進み、ＡＬＵ部２２は、プロセスパケットに配置されている注目ブロックの画素それぞれと、その画素に対応する候補ブロックの画素それぞれとの差分絶対値を計算し、さらに、その総和を計算する。そして、ＡＬＵ部２２は、その差分絶対値の総和と、プロセスパケットの差分絶対値和部に配置されている差分絶対値和とを加算し、その加算値を、新たな差分絶対値和として、プロセスパケットの差分絶対値和部に上書きする。

【０１１８】

なお、ステップＳ２６において、ＡＬＵ部２２は、プロセスパケットに配置された注目ブロックと候補ブロックの画素のうちの、差分絶対値の計算に用いたものは、それ以降の差分絶対値の計算に不要であるため、例えば、プロセスパケットから削除する。

【０１１９】

以上のように、ステップＳ２６において、新たな差分絶対値和が、プロセスパケットの差分絶対値和部に上書きされることにより、差分絶対値和演算命令の実行は終了し、ステップＳ１４に進む。

【０１２０】

ステップＳ１４では、ＡＬＵ部２２は、ステップＳ１３で行った処理に対応して、必要に応じて、プロセスパケットの状態部を書き換え、ステップＳ１５に進む。ステップＳ１５では、ＡＬＵ部２２は、プロセスパケットを、後段の演算処理ユニット2_{n+1}に送信し、処理を終了する。

【０１２１】

プロセス生成部１において、差分絶対値和演算命令を有するプロセスパケットが生成され、このプロセスパケットが、演算処理ユニット2_1乃至2_{12}を順次転送されることにより、演算処理ユニット2_1乃至2_{12}それぞれでは、図１３のフローチャートにしたがったプロセス実行処理が行われる。これにより、プロセスパケットが、演算処理ユニット2_1乃至2_{12}を順次転送されていく過程で、注目ブロックと候補ブロックの差分絶対値和が、図１４および図１５に示すように求められる。

【０１２２】

即ち、いま、図１４に示すように、メモリ12_nに、注目ブロックの一部が、メモリ12_{n+1}に、注目ブロックの残りと候補ブロックの一部が、メモリ12_{n+2}に、候補ブロックの残りが、それぞれ記憶されているものとする。また、演算処理ユニット2_nのＰＥ１１を、以下、適宜、$PE11_n$と表すこととする。

【０１２３】

この場合、演算処理ユニット2_nでプロセスパケットが受信されると、$PE11_n$は、メモリ12_nに記憶されている注目ブロックの一部を読み出し、プロセスパケットに配置して、次段の演算処理ユニット2_{n+1}に転送する。

【０１２４】

ここで、$PE11_n$では、図１５に示すプロセスパケットが送受信される。

【０１２５】

なお、図１５では、図が煩雑になるのを避けるため、プロセスパケットについて、図１５Ａに示すように、その先頭から、ＰＩＤ部、注目ブロック読み出し状態部、候補ブロック読み出し状態部、命令部、注目ブロックアドレス部、候補ブロックアドレス部、注目ブロックデータ部、候補ブロックデータ部、差分絶対値和部だけを図示してある。

【０１２６】

注目ブロックおよび候補ブロックは、図１４で説明したように、メモリ12_n乃至12_{n+2}に記憶されているから、プロセスパケットが、$PE11_n$で受信される前は、注目ブロックおよび候補ブロックのいずれの読み出しを行われておらず、従って、注目ブロック

読み出し状態部と、候補ブロック読み出し状態部の状態情報は、図１５Ｂに示すように、いずれも「未」になっている。さらに、この場合、注目ブロックアドレス部と候補ブロックアドレス部には、それぞれ、注目ブロックのアドレスａｄｄ１と候補ブロックのアドレスａｄｄ２がセットされている。また、差分絶対値和部には、初期値としての０がセットされている。

【０１２７】

なお、命令部には、差分絶対値和演算命令を表す「ＭＥ」がセットされている。

【０１２８】

以上のようなプロセスパケットが、ＰＥ１１$_n$で受信され、図１４で説明したように処理されることにより、次のようなプロセスパケットが、ＰＥ１１$_n$からＰＥ$_{n+1}$に転送される。

【０１２９】

即ち、ＰＥ１１$_n$からＰＥ$_{n+1}$に転送されるプロセスパケットには、図１５Ｂに示すように、メモリ１２$_n$に記憶されている注目ブロックの一部の画素ｄａｔａ＿ａ１が、新たに配置される。さらに、ＰＥ１１$_n$では、メモリ１２$_n$から注目ブロックの一部の画素ｄａｔａ＿ａ１が読み出されたことから、プロセスパケットの注目ブロック読み出し状態部の状態情報が、「未」から「中」に書き換えられる。

【０１３０】

演算処理ユニット２$_{n+1}$が、演算処理ユニット２$_n$からのプロセスパケットを受信すると、ＰＥ１１$_{n+1}$は、図１４に示すように、メモリ１２$_{n+1}$に記憶されている注目ブロックの残りと、候補ブロックの一部を読み出す。ここで、演算処理ユニット２$_n$からのプロセスパケットには、注目ブロックの一部が配置されているから、ＰＥ１１$_{n+1}$は、メモリ１２$_{n+1}$から読み出した注目ブロックの残りとあわせて、注目ブロック全体の画素を取得することになる。

【０１３１】

ＰＥ１１$_{n+1}$は、注目ブロック全体の画素と、メモリ１２$_{n+1}$から読み出した候補ブロックの一部の画素とを用いて計算可能な差分絶対値和を求め、プロセスパケットに配置する。さらに、ＰＥ１１$_{n+1}$は、差分絶対値和の演算に用いられなかった注目ブロックの画素を、プロセスパケットに配置し、次段の演算処理ユニット２$_{n+2}$に転送する。

【０１３２】

即ち、ＰＥ１１$_{n+1}$からＰＥ$_{n+2}$に転送されるプロセスパケットには、図１５Ｂに示すように、注目ブロックの画素のうち、差分絶対値和の演算に用いられなかった画素ｉａｔａ＿ａ２が、注目ブロックデータ部の画素ｉａｔａ＿ａ１に代えて配置されるとともに、ＰＥ$_{n+1}$で求められた差分絶対値和ｓｕｍ１が、差分絶対値和部の初期値０に代えて配置される。さらに、ＰＥ１１$_{n+1}$では、注目ブロックの画素のすべてが取得されるとともに、候補ブロックの一部の画素が取得されたことから、プロセスパケットの注目ブロック読み出し状態部の状態情報が、「中」から「終」に書き換えられるとともに、候補ブロック読み出し状態部の状態情報は、「未」から「中」に書き換えられる。

【０１３３】

演算処理ユニット２$_{n+2}$が、演算処理ユニット２$_{n+1}$からのプロセスパケットを受信すると、ＰＥ１１$_{n+2}$は、図１４に示すように、メモリ１２$_{n+2}$に記憶されている候補ブロックの残りを読み出す。

【０１３４】

ＰＥ１１$_{n+2}$は、メモリ１２$_{n+2}$から読み出した候補ブロックの残りの画素と、プロセスパケットに配置されている注目ブロックの画素とを用いて計算可能な差分絶対値和を求め、プロセスパケットに配置されている差分絶対値和と加算する。そして、ＰＥ１１$_{n+2}$は、その加算値を、新たな差分絶対値和として、プロセスパケットに上書きする形で配置し、次段の演算処理ユニット２$_{n+3}$に転送する。

【０１３５】

即ち、ＰＥ１１$_{n+2}$では、注目ブロックと候補ブロックのすべての画素についての差分

絶対値和ｓｕｍ２が求められるから、ＰＥ１１$_{n+2}$からＰＥ$_{n+3}$に転送されるプロセスパケットには、図１５Ｂに示すように、その差分絶対値和ｓｕｍ２が、差分絶対値和部の差分絶対値和ｓｕｍ１に代えて配置される。また、プロセスパケットにおいて注目ブロックデータ部に配置されていた注目ブロックの画素ｉａｔａ＿ａ２は、すべて、差分絶対値和の計算に用いられるから、注目ブロックデータ部から、注目ブロックの画素ｉａｔａ＿ａ２が削除される。さらに、ＰＥ１１$_{n+2}$では、候補ブロックの残りの画素が取得されたことから、プロセスパケットの候補ブロック読み出し状態部の状態情報が、「中」から「終」に書き換えられる。

【０１３６】

この場合、プロセスパケットが、ＰＥ１１$_{n+2}$から出力される時点で、そのプロセスパケットには、注目ブロックと候補ブロックのすべての画素についての差分絶対値和ｓｕｍ２が配置されている。従って、演算処理ユニット２$_{n+3}$以降では、プロセスパケットは、単に転送されていき、最終的に、集計部３（図５）で受信される。

【０１３７】

プロセス生成部１では、図８のステップＳ４乃至６の処理によって、注目ブロックの探索範囲内に選択しうる候補ベクトルすべてについて、差分絶対値和を演算するための差分絶対値和演算プロセスが生成されるが、このプロセスに対応するプロセスパケットすべてが集計部３で受信されると、集計部３は、注目ブロックの探索範囲内に選択しうる候補ベクトルすべてについて、差分絶対値和を取得する。

【０１３８】

プロセス生成部１では、図８のステップＳ４乃至６の処理の後、ステップＳ７およびＳ８において、最小値判定プロセスが生成され、最小値判定命令を含むプロセスパケットが出力される。このプロセスパケットは、演算処理ユニット２$_{1}$乃至２$_{1_2}$を経由して、集計部３で受信される。集計部３は、最小値判定命令を含むプロセスパケットを受信すると、それまでに受信した、差分絶対値和演算プロセスに対応するプロセスパケットから、最小の差分絶対値和が配置されているものを選択し、さらに、そのプロセスパケットに配置されている候補ベクトルを、注目ブロックの動きベクトルとして出力する。

【０１３９】

以上のように、ＰＥ１１$_{n}$とメモリ１２$_{n}$からなる複数の演算処理ユニット２$_{n}$を一次元的に接続し、その複数の演算処理ユニット２$_{n}$において、命令と必要なデータが配置されたプロセスパケットを転送させることにより、動きベクトルを検出するための処理を行うようにしたので、長いデータバスを駆動する際の問題としての、例えば配線遅延や配線間のクロストーク、反射の影響などを回避することが可能となる。

【０１４０】

即ち、ＰＥ１１$_{n}$とメモリ１２$_{n}$との間や、演算処理ユニット２$_{n}$どうしを結ぶデータバスは、短くて済むので、長いデータバスを駆動する際の問題を回避することができる。従って、動きベクトル装置の設計にあたり、長いデータバスを駆動する際の問題を考慮する必要がないので、設計容易なハードウェアによって、動きベクトルを検出することができる。

【０１４１】

さらに、図５のベクトル検出装置では、プロセスパケットが、一次元的に接続された複数の演算処理ユニット２$_{n}$を順次転送されていくので、プロセス生成部１は、先に送信したプロセスパケットが、集計部３に到達する前に、次のプロセスパケットを送信することができる。この場合、複数の演算処理ユニット２$_{n}$それぞれでは、その演算処理ユニット２$_{n}$で受信されたプロセスパケットが処理されるから、容易に、並列処理が実現されることになる。

【０１４２】

また、図５のベクトル検出装置では、複数の演算処理ユニット２$_{n}$が一次元的に接続されているので、例えば、１つの演算装置に、複数のメモリを接続する場合に比較して、演算処理ユニット２$_{n}$どうしを結ぶデータバスのバス幅を短くすることができる。即ち、１つ

の演算装置に、複数のメモリから同時にデータを転送しようとする場合には、データのビット数を、メモリの個数倍した値のバス幅が必要となる。これに対して、複数の演算処理ユニット2_nを一次元的に接続し、プロセスパケットを転送させる場合には、演算処理ユニット2_nどうしを結ぶデータバスのバス幅は、１つのプロセスパケットを転送するのに必要な分だけで足りる。

【０１４３】

なお、本実施の形態では、複数の演算処理ユニット2_nを一次元的に接続するようにしたが、複数の演算処理ユニット2_nは、その他、例えば、図１６に示すように、いわば格子状に接続すること等も可能である。

【０１４４】

また、本実施の形態では、図６に示したように、メモリ１２を、５つのバンクに分割し、５フィールドの画像データを、別々のバンクに記憶することができるようにしたが、メモリ１２は、その他、例えば、図１７に示すように、２つのバンクに分割し、一方のバンクには、注目フィールドの画像データを記憶させ、他方のバンクには、参照フィールドの画像データを記憶させるようにすること等が可能である。

【０１４５】

さらに、本実施の形態では、集計部３において、ある注目ブロックについて、探査範囲内で選択しうるすべての候補ブロックとの差分絶対値和が求められた後に、その中から、最小の差分絶対値和を求めるようにしたが、その他、集計部３では、ある注目ブロックについて、１つの候補ブロックとの差分絶対値和が求められるたびに、その差分絶対値和が、いままでに求められた差分絶対値和の中で最小のものかどうかを判定するようにすることが可能である。

【０１４６】

なお、ある注目ブロックについて、探査範囲内で選択しうるすべての候補ブロックとの差分絶対値和が求められた後に、その中から、最小の差分絶対値和を求める場合には、すべての候補ブロックとの差分絶対値和を記憶しておくための大きな容量のメモリが必要となるが、シーケンスは簡単となり、また、演算処理ユニット2_nの間を伝送されるデータも少なくなる。

【０１４７】

一方、ある注目ブロックについて、１つの候補ブロックとの差分絶対値和が求められるたびに、その差分絶対値和が、いままでに求められた差分絶対値和の中で最小のものかどうかを判定する場合には、シーケンスが複雑になり、演算処理ユニット2_nの間を伝送されるデータが多くなるが、差分絶対値和を記憶しておくためのメモリは、小さな容量のもので済む。

【０１４８】

ここで、本明細書においてフローチャートを参照して説明した処理ステップは、必ずしもフローチャートとして記載された順序に沿って時系列に処理する必要はなく、並列的あるいは個別に実行される処理（例えば、並列処理あるいはオブジェクトによる処理）も含むものである。

【０１４９】

なお、上述した動きベクトル検出装置は、ＭＰＥＧ方式により画像をエンコードするＭＰＥＧエンコーダの他、動きベクトルの検出を行うあらゆる装置に適用可能である。

【０１５０】

【発明の効果】

以上の如く、本発明によれば、設計容易なハードウェアによって、動きベクトルを検出することが可能となる。

【図面の簡単な説明】

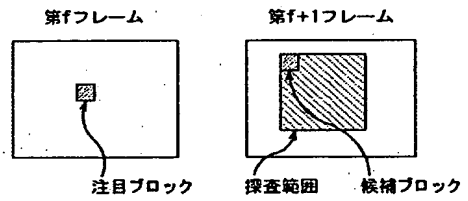
【図１】ブロックマッチング法を説明する図である。

【図２】ブロックマッチング法を説明する図である。

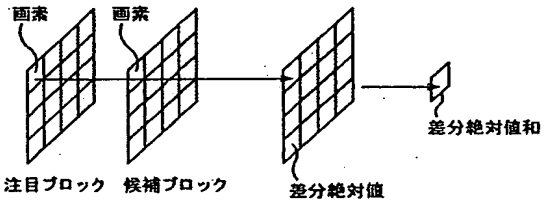
【図３】従来の動きベクトル検出装置の一例の構成を示すブロック図である。

【図４】従来の動きベクトル検出装置の他の一例の構成を示すブロック図である。
【図５】本発明を適用した動きベクトル検出装置の一実施の形態の構成例を示すブロック図である。
【図６】演算ユニット２$_n$の構成例を示すブロック図である。
【図７】プロセスパケットのフォーマットを示す図である。
【図８】プロセス生成処理を説明するフローチャートである。
【図９】プロセス実行処理を説明するフローチャートである。
【図１０】画像の書き込みを行うプロセス実行処理を説明するフローチャートである。
【図１１】メモリ１２$_n$に画像データが書き込まれた状態を示す図である。
【図１２】画像の読み出しを行うプロセス実行処理を説明するフローチャートである。
【図１３】差分絶対値和を演算するプロセス実行処理を説明するフローチャートである。
【図１４】演算処理ユニット２$_n$の処理を説明するための図である。
【図１５】プロセスパケットの変化を示す図である。
【図１６】演算処理ユニット２$_n$どうしの接続方法を示す図である。
【図１７】メモリ１２におけるバンクの構成方法を示す図である。
【符号の説明】
１　プロセス生成部，　２，２$_1$乃至２$_{12}$　演算処理ユニット，　３　集計部，　１１，１１$_1$乃至１１$_{12}$　ＰＥ，　１２，１２$_1$乃至１２$_{12}$　メモリ，　２１　フリップフロップ，　２２　ＡＬＵ部

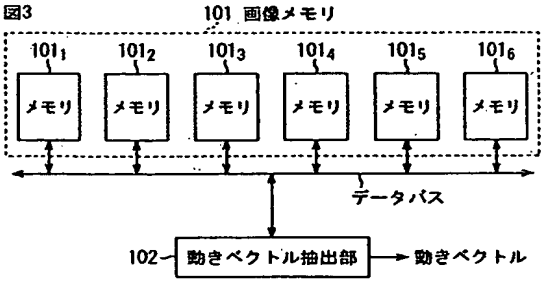
【図１】
図1



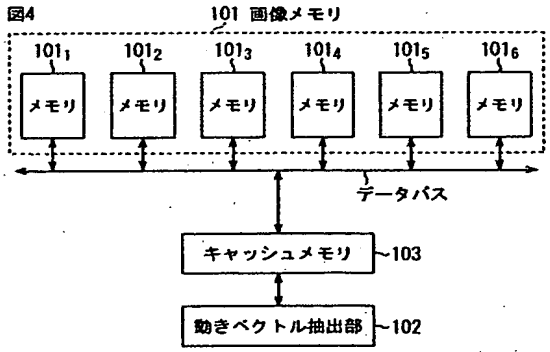
【図２】
図2



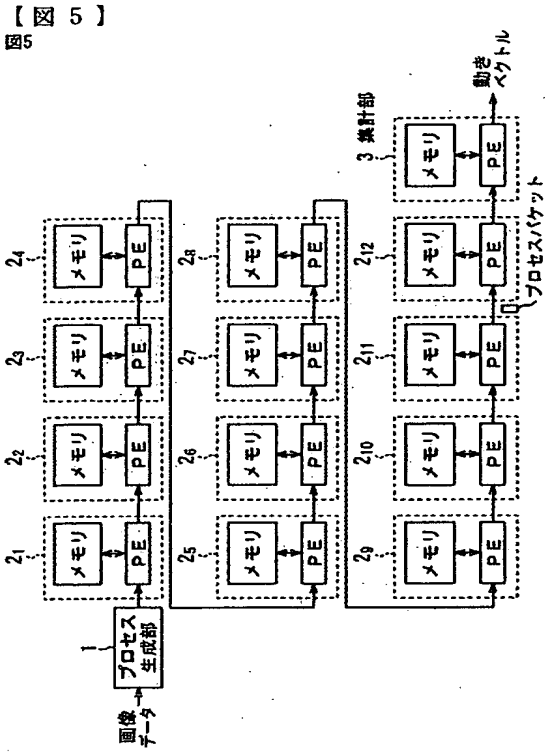
【図３】
図3



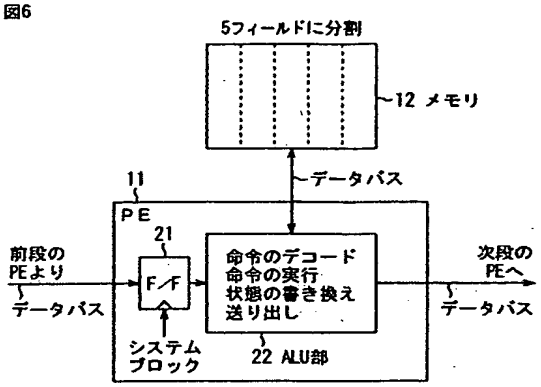
【図４】
図4



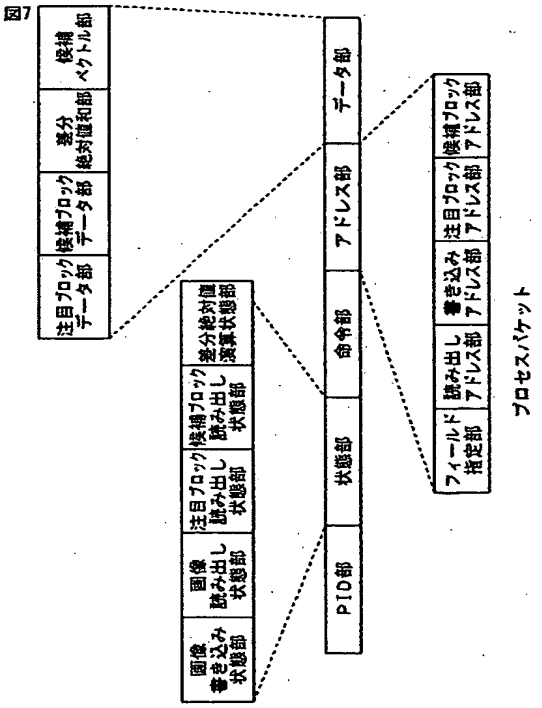
【図５】
図5



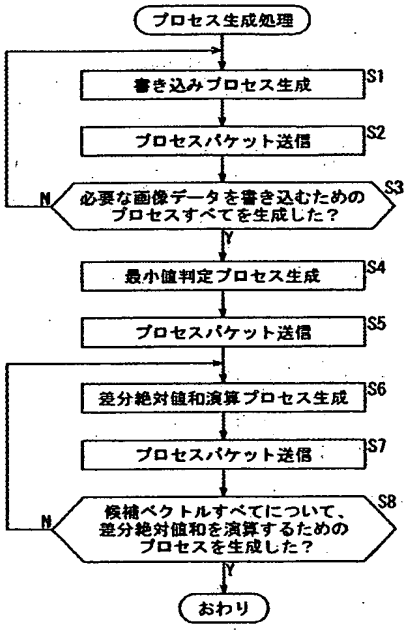
【図６】
図6



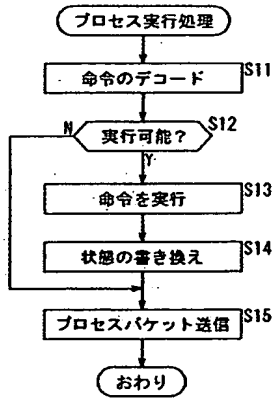
【図７】
図7



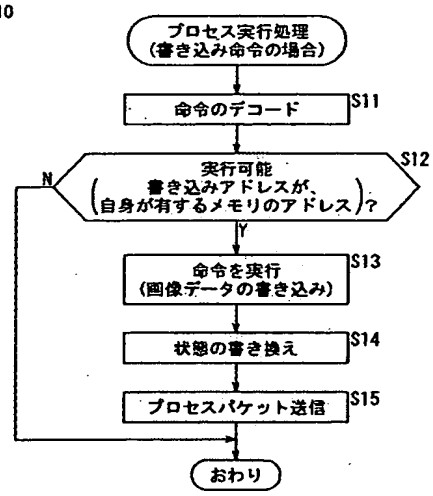
【図８】
図8



【図９】
図9



【図１０】
図10



【図１１】
図11

図11A

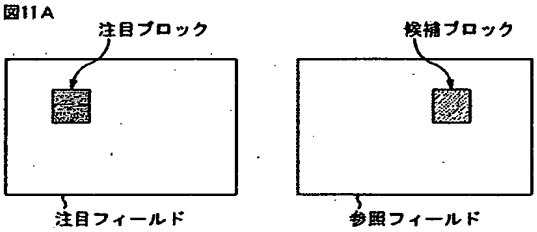
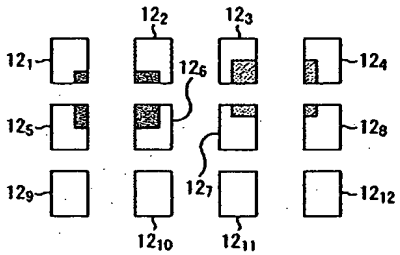
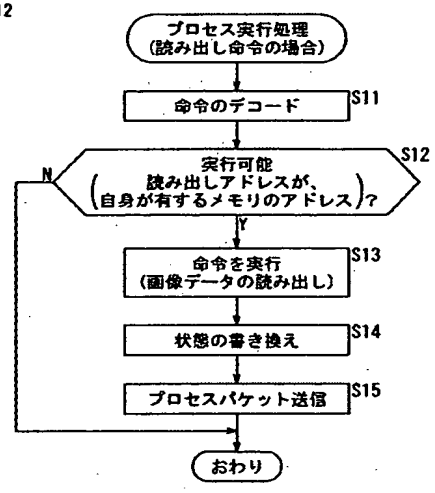


図11B

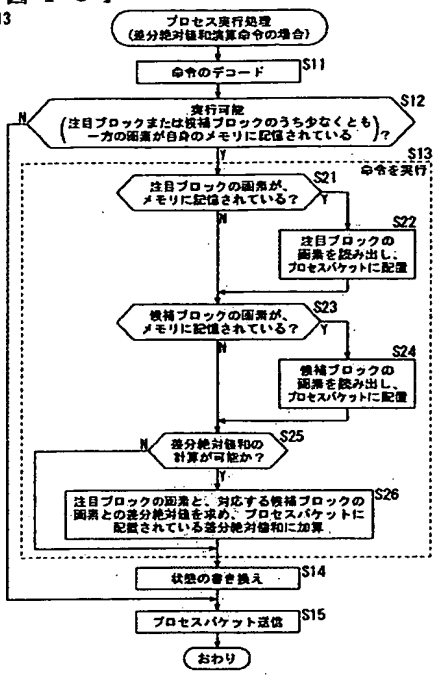


【図１２】
図12



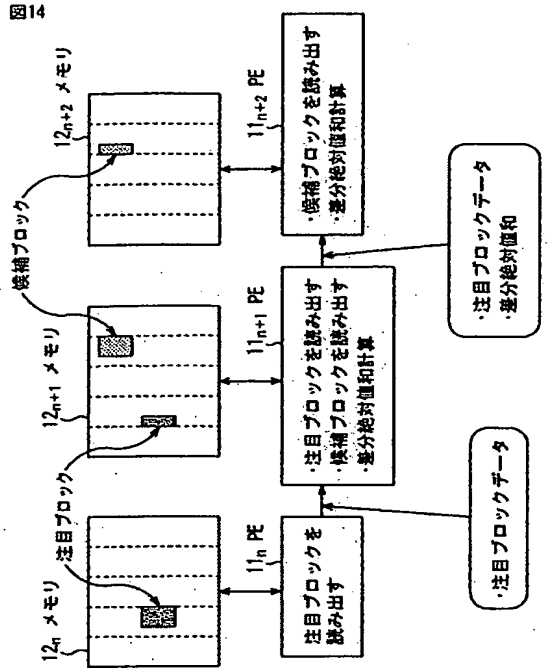
【図１３】

図13



【図１４】

図14



【図１５】

図15

図15A

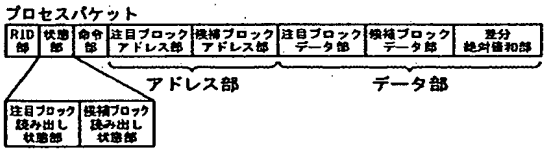
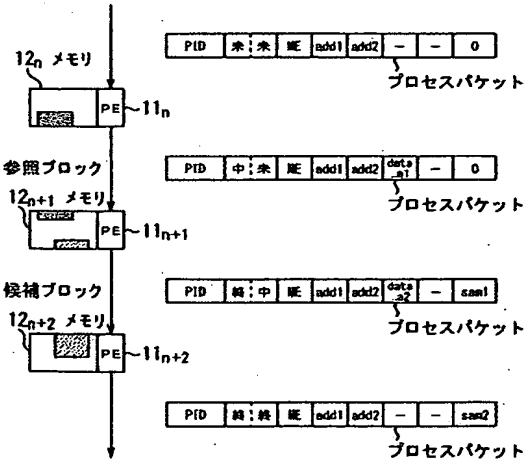
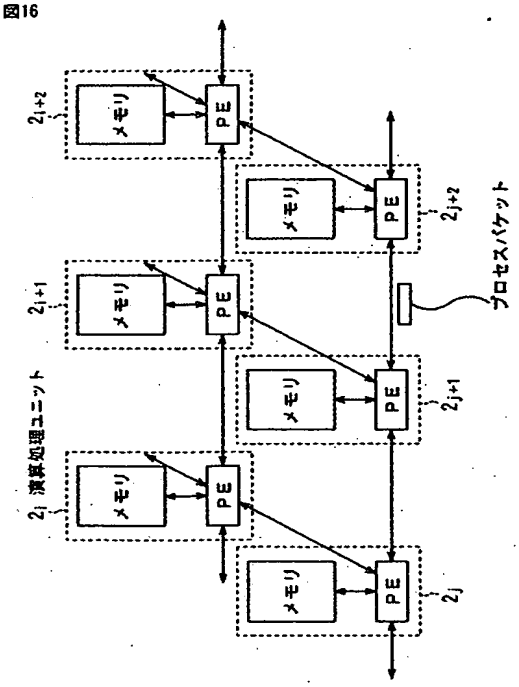


図15B

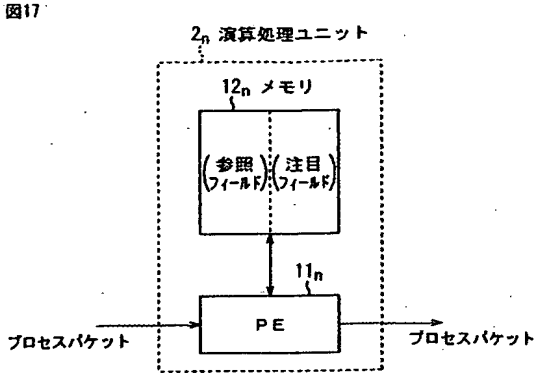


【図１６】

図16



【図１７】
図17


2n 演算処理ユニット
12n メモリ
参照フィールド
注目フィールド
11n
PE
プロセスパケット
プロセスパケット

フロントページの続き

Fターム(参考) 5J064 AA01 BA13 BB01 BC01 BC03 BC08 BC14 BD01